12/24/02 JC13 U.S. PTO

60435870 122402

A/Prov

JC13 U.S. PTO 60/435870 12/24/02

PROVISIONAL APPLICATION FOR PATENT COVER SHEET

This is a request for filing a PROVISIONAL APPLICATION FOR PATENT under 37 CFR 1.53(b)(2).

INVENTOR(s)/APPLICANT(s)

Given Name (first and middle [if any])	Family Name or Surname	Residence (CITY AND EITHER STATE OR FOREIGN COUNTRY)
Prem	YADAV	Manassas, VA

☐ Additional inventors are being named on the ___ separately numbered sheets attached hereto.

TITLE OF THE INVENTION (280 characters max)

Meta-Search Engine with Filtering

CORRESPONDENCE ADDRESS

☒ Customer Number: **6449**

☐ Firm or Individual Name	Rothwell, Figg, Ernst & Manbeck, P.C.				
Address	1425 K Street, N.W.				
Address	Suite 800				
City	Washington	State	D.C.	ZIP	20005
Country	U.S.A.	Telephone	202-783-6040	Fax	202-783-6031

ENCLOSED APPLICATION PARTS *(check all that apply)*

☒ Specification *Number of Pages* [15]
☒ *Number of Figures* [8]
☐ Application Data Sheet. See 37 CFR 1.76
☐ CD(s), Number _____
☐ Other *(specify)* _____

METHOD OF PAYMENT OF FILING FEES FOR THIS PROVISIONAL APPLICATION FOR PATENT *(check one)*

☒ Applicant claims small entity status. See 37 CFR 1.27
☒ A check or money order is enclosed to cover the filing fee
☐ The Commissioner is hereby authorized to charge filing fees or credit any overpayment to Deposit Account Number: 02-2135
☐ Payment by credit card. Form PTO-2038 is attached.

Filing Fee Amount: $80.00

The invention was made by an agency of the United States Government or under a contract with an agency of the United States Government.

☒ No.

☐ Yes, the name of the U.S. Government agency and the Government contract number are: _____

Respectfully submitted,

SIGNATURE Martin M. Zoltick Date 12/24/02

TYPED or PRINTED NAME Martin M. Zoltick
TELEPHONE : 202-783-6040

REGISTRATION NO. 35,745
Docket Number: 2601-108

Principal Investigator/Program Director (Last, First, Middle): Yadav, Prem

DESCRIPTION: State the application's broad, long-term objectives and specific aims, making reference to the health relatedness of the project. Describe concisely the research design and methods for achieving these goals. Avoid summaries of past accomplishments and the use of the first person. This abstract is meant to serve as a succinct and accurate description of the proposed work when separated from the application. If the application is funded, this description, as is, will become public information. Therefore, do not include proprietary/confidential information. **DO NOT EXCEED THE SPACE PROVIDED.**

We will develop a secure, focused and specialized knowledge management tool: **Xactans** (exact answer, combined) for finding project- specific scientific information from flat files, scientific literature and biological databases currently accessible via the Internet. **Xactans** is a multi-tier search system that will allow users to obtain the information they actually want using a single query profile. Unlike currently used search engines, **Xactans** will delve into scientific databases and probe PDF format files--a file format widely used in scientific disciplines. **Xactans** will employ user-assisted artificial intelligence and association discovery tools with a profile-based information filtering mechanism. For the first time, users will have the option to assign weight to key terms in their query profile, which will be used to rank output documents and increase the probability of user satisfaction. **Xactans** will download, analyze each document and display results that include helpful hints regarding the contents of each article. Thus, users will not need to read articles in their entirety to evaluate their relevance. Search results can be stored in an electronic notebook that will feature the option of creating a file cabinet stack of categorized folders comprising a virtual private datastore. **Xactans** will provide higher quality query results than currently available search engines including PubMed and Google, and will eventually be used as the preferred scientific query tool.

PERFORMANCE SITE(S) *(organization, city, state)*

Prem Yadav LLC
10801 University Boulevard
Manassas, Virginia 20110-2209

KEY PERSONNEL. See instructions. *Use continuation pages as needed* to provide the required information in the format shown below. Start with Principal Investigator. List all other key personnel in alphabetical order, last name first.

Name	Organization	Role on Project
Yadav, Prem, Ph.D	Prem Yadav LLC	Principal Investigator
Ding, Yan, M.D.	Prem Yadav LLC	Investigator
Nguyen, Truc, M.S.	Prem Yadav LLC	Investigator
Piselli, Anthony, Jr., M.A.	Prem Yadav LLC	Investigator
Ravich, Vadim, M.S.	Prem Yadav LLC	Investigator

Disclosure Permission Statement. Applicable to SBIR/STTR Only. See instructions. ☐ **Yes** ☐ **No**

A. Specific Aim:

Development of a scientific search and knowledge management tool: **Xactans** that can search project specific scientific information from the World Wide Web, journal articles, literature databases and integrated molecular biology databases. In Phase one, we propose to (i) develop the Knowledge Pack module, (ii) apply **Xactans** to .edu domain sites and journal articles, available on PubMed Central, for search, retrieval and document processing, (iii) write software for the **EScore** scoring matrix and (iv) integrate the **ESpider** database search module. Completion of **Xactans** is anticipated during Phase two, and release will occur in Phase three.

B. Background and Significance

Recent years have seen explosive growth in the number and content of vital biological databases that contain essential information regarding structural biology, genomics, proteomics, metabolic and signal transduction pathways, clinical trial results, chemical structures, and Patents—both applied and granted (Wilkinson, '02, Voss, '01). The ability of the scientific community to access this information relies almost completely upon well-established search engines such as PubMed, the US Patent and Trademark Office (USPTO) web page, and Google. Many individual publishers have designed their own search engines such as Elsevier Sciences ScienceDirect, and Wiley InterSciences service, but these are of extremely limited scope. Unfortunately, a user-friendly search engine capable of providing a single portal with sufficient reach to provide, exclusively, desired information to the research community has yet to be introduced. Moreover, while preparing this proposal we learned that the Department of Energy shut down the public domain resource "PubScience" that cross indexed nearly 2 million government reports and academic articles. Most of the private scientific search engines such as SciRus, SciFinder, and Science4Search, access online resources and their own databases. Thus, they cannot be customized based on site licenses of user institutions or individual subscribers. **Xactans** will be user customized.

The most commonly used search engines only provide access to a fraction of the desired information. For example, to obtain basic information regarding genomes, primary nucleotide, or amino acid sequence and protein structural data, a user might query National Center for Biotechnology Information (NCBI) databases. However, a more informed user might also query other databases: e.g. the Stanford Microarray database, PlasmoDB at the University of Pennsylvania, the metabolic pathway database at Yale University, Structural Classification of Protein (SCOP) at Cambridge, UK, the Nucleic Acid Data Bank (NDB) and Protein Data Bank (PDB) at Rutgers University, Signaling Pathway database (SPAD) and DNA database of Japan, the Transgenic and Targeted Mutant Animal Database (TBASE) at John Hopkins, Clintrials clinical studies database, and the USPTO database--just to name a few. Existing autonomous, biological databases contain related data that are more valuable when interconnected. However, it is currently not possible to simultaneously query related data because source databases are built by different teams, in different locations, for different purposes, and are comprised of different database architectures and design. To obtain desired information, rigorous scientists must query multiple remote or local heterogeneous data sources, and manually integrate retrieved data without the aid of intelligent data analysis and visualization tools. **Xactans** will provide users access to customized and integrated Internet-accessible databases via one portal of entry, such that queries need not be repeated multiple times in order to obtain needed information.

Currently available search engines typically input keywords or phrases that are heavily supported by Boolean logic terms such as "and", "not", and "or". They monitor and can rank query output based on hit frequencies or chronology such that more recent database inputs, or popular links, as determined by the user community, appear first. Output can appear ranked by hyperlink pattern, independent of precise search specifications. This is based on the assumption that important web pages are likely to be those that have relatively numerous links to other pages, or are frequently linked from other pages. Unfortunately, current ranking schemes often provide the desired output along with too much undesired output (Lawrence and Giles, 1998). Thus, users must scan query output manually to find what they need. **Xactans** will harness a systematic dynamic query profiler, feature scoring, and display of retrieved documents via a knowledge-based system that facilitates user editing. Thus, **Xactans** will aid users such that less time and effort need be applied in order to more exclusively obtain precisely, the desired information.

There is a critical need to be able to maintain current and updated information regarding topics of interest to individual users. Moreover, scientific investigators have aligned themselves into specialized areas, and might benefit from a search engine capable of enlarging their peripheral vision. In the event that

specialized queries are repeated over time, **Xactans** will offer the scientific community the ability to maintain their own datastore, in private accounts, that contain specialized information, and furthermore, enable users to more easily encounter supplemental information of direct relevance to their original query.

RESEARCH DESIGN AND METHODS

Xactans Internal Structure:

Xactans will be a web-based, enterprise-wide knowledge management application that will use dynamic search profiles (**DSP**) to identify relevant information from a wide range of sources including journal article databases and web pages. In addition, query terms will be routed directly into scientific databases via a separate database query module.

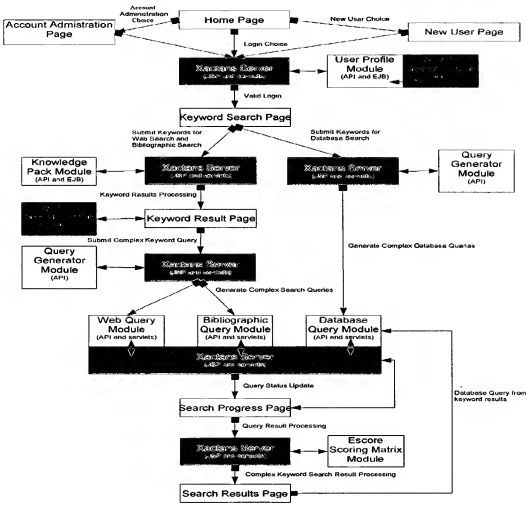

Figure 1. Operations Flow Diagram of Xactans. Modules in gray (light blue) boxes constitute the **ESpider** system.

Access to this information will depend upon the user environment, and be provided via a user interface that will allow users to annotate, organize, and share their data. A backend repository will allow users to store all relevant information locally or as links to the original source. User repositories will be configurable for individuals or organizations.

The following operational components will be developed:

- **ESpider**: A customized scientific indexing, search, retrieval and analysis system responsible for searching, retrieving, processing, and ranking documents based on relevance.

- **EScore**: A dynamic document assessment module, which executes complex keyword searches, and accepts selected parameters from the **ESpider's** query result processing. **EScore** will employ an interconnected scoring matrix system to assess term relatedness, because a query term might be associated with other, companion and synonymous terms. The **EScore** module will also provide rank information for each retrieved document. **ESpider** will use this information to filter the document list, and display query results in descending order based on relevance ranking.

- The Knowledge Pack (**NP**): A collection of commonly used scientific terms in life science disciplines (Biochemistry, Cell Biology, Immunology, Molecular Biology, Physiology, etc.) based on 871,584 concepts, and 2.1 million concept names extracted from the Unified Medical Language System (UMLS)—a project of the National Library of Medicine (NLM), from an electronic encyclopedia and scientific dictionary available in the public domain. The information in the **NP** will be rearranged into a customized database setting, and will be continuously updated with the help of life scientists because all searches will be logged, (to protect user privacy, only words will be extracted without attribution to user information) and a list of frequently used and missing terms will be checked. **NP** terms will include the definition, synonyms, and associated words typically found in context with a potential query term. In addition, **Xactans** users will contribute by providing definitions, acronyms, synonyms and related query terms. All entries will be cross-referenced to each other and to related resources on the Internet.

- The Dynamic Search Profile (**DSP**): This is a user-assisted query profile generating system that works with the NP for modulating query terms and assigning weights to establish a hierarchical output.

- It is our intention that **ESpider** will selectively analyze scientific documents. During Phase I, documents containing at least 5 scientific terms from the **NP** will be classified as scientifically relevant, and this minimal criterion will be further optimized during the development process.

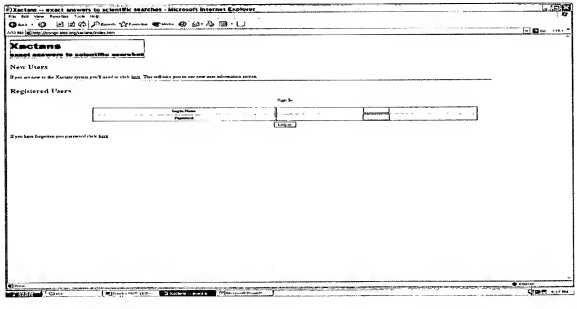

Figure 2. Snapshot of the user login window.

Xactans External Structure:

Graphical user interfaces will be developed to interact with users with applications that mimic popular and familiar features commonly found on desktop computers. **Xactans** will employ a web based client interface that will provide for query input, weighting, and source navigation via a set of hyperlinks to be displayed in a

separate frame. The home page frame will initially display the data for the current function. Results of searches are displayed in a third frame that is initially hidden until the result display is needed. A desktop application provides the same set of functionality but through a familiar menu based system.

Xactans execution:

Following user registration/authentication (Figure 2) and query input (Figure 3), **Xactans** will offer users the option of creating a **DSP** in order to help them tailor the focus of their search.

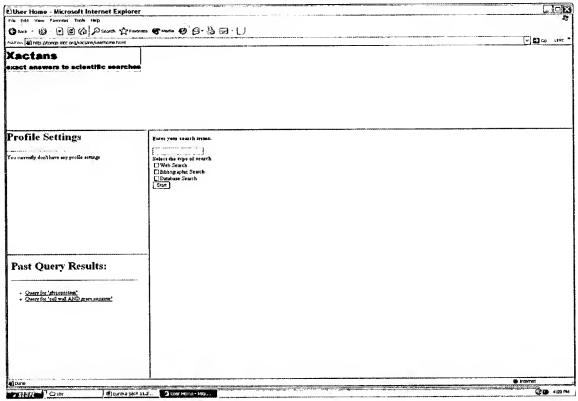

Figure 3. Snapshot of the user profile, search module options and query entry.

The **DSP** will employ the **NP** and guide users to pick and choose appropriate terms in their search by presenting a list of synonyms and related terms for better definition of the search field. For example, suppose a user would like information regarding the crystal structure of **HIV** reverse transcriptase in order to develop a better inhibitory drug. They would access the **Xactans** home page on the Internet through proper authentication, including site licensing check, and proceed to input the query: "reverse transcriptase". Figure 4 shows the data retrieved from the **NP.** Thereafter users will be invited to establish the **DSP** by click entering included synonyms, related terms, and weights. In this case the user is interested in articles on the crystal structure of reverse transcriptase. The user will enter crystal and structure as user-defined related words.

A Boolean query **DSP** for "reverse transcriptase" augmented by the user-selected terms will be used by the **ESpider** module for the initial search as illustrated above. The returning documents will be processed by **ESpider**, and the **EScore** module will be activated.

Let us say that there are 5 keywords and 9 related words found in the abstract of the first document, in addition there are 23 keywords and 29 related words found in the rest of the document. There are 2 keywords and 6 related words found in the abstract of the second document, 40 keywords and 45 related words found in the rest of the document. The rating matrix of keywords found in the document vs. related words found in the same document (Figure 5) is serially positioned to the matrix of keywords found in the abstract vs. related

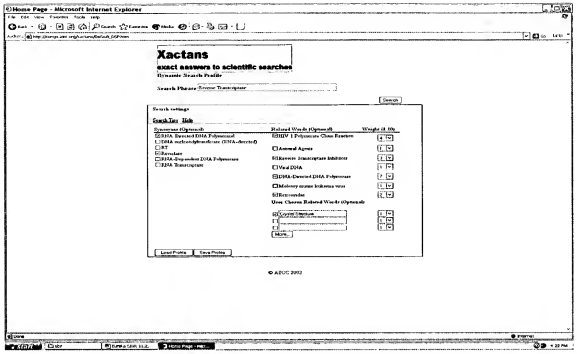

Figure 4. Snapshot of the DSP with assignment of weight entry for selected terms.

words found in the same abstract allowing us to add the scores of each matrix together. The score is calculated for the first and second documents as follows:

First document score = M1[3,3]+M2[2,2]=9.7+2.6=12.3
Second document score = M1[1,2]+M2[3,3]=2.5+6.7=9.2

Number of Related Words in the Abstract

Number of Keywords in the Abstract

M1

	<3	2> <5	4> <10	>9
<4	0.2	0.3	2.0	2.5
3> <7	**2.5**	2.6	4.7	5.2
6> <11	5.0	5.0	**9.7**	10.3
>10	10.0	12.0	13.0	15.0

Number of Related Words in the Document

Number of Keywords in the Document

M2

	<15	14> <25	24> <51	>50
<17	0.2	0.3	1.0	3.5
16> <30	2.5	**2.6**	3.7	6.2
29> <60	5.0	5.0	**6.7**	13.3
>59	10.0	12.0	11.0	17.0

Figure 5. Two-dimensional matrices M1 and M2 producing a specific score of an instance for keywords

Based on the assessment of two documents using two relational matrices in series, it turns out that the first document with the score of 12.3 will be ranked higher than the second document with the score of 9.2 showing a preference for the words in the abstract in this case.

The user will thereafter have the ability to select documents of interest and peruse desired output and/or save query results in personal profiles. Users will have the option of also displaying summaries of retrieved documents, which can also indicate the degree of relevance to the user. With a small amount of practice, users will become accustomed to scanning the hit frequency of desired terms, and thereby gain insights regarding the content of retrieved documents without the need of actually reading them in their entirety.

To enable users to maintain up-to-date information regarding topics of interest, past queries can be readily retrieved from their personal datastore, modified if needed, and previously used DSPs can be resubmitted. The option to save retrieved documents in HTML or pdf format will be available.

Operations Design and Architecture:

A multi-tiered architecture will be implemented using a Java 2 Platform Enterprise Edition (J2EE) compatible code base in the enterprise system. The outermost, client tier will provide a customizable graphical user interface, which communicates with the middle tier through an Application Programmer Interface (API). APIs will be used to obtain access to needed services such as web searches, online bibliographic/ journal database searches, integrated online database searches and document management. Middle-tier operations include the **Xactans** web application--which interacts with the client tier—handles user requests and provides server processing operations, analysis tools, data display tools, query building engines, user session management, and security services. This web application sits above a database management system on the bottom tier. The communication between the web application and the bottom tier database is abstracted into internal APIs allowing different database systems to be used for the repository (Figure 6).

Middle tier applications:

The server side web application (**Xactans** Server) is composed of numerous J2EE components: Servlets, JavaServer Pages (JSPs), and Enterprise Java Beans (EJBs), which will be used to coordinate user requests, the three search services, connections to databases, and document management. Initial requests are handled by servlets or JSPs. These services, combined, utilize EJBs to communicate to data and service sources. In anticipation of large amounts of user requests and large analysis processes the web application will be developed with fail-over and redundancy in mind in order to scale to demand. Techniques such as clustering, load balancing, and request queues will be incorporated into the web applications' development.

The **NP**:

An example of the above mentioned J2EE component interaction is the initial search term lookup, which is first handled by a servlet, which then utilizes EJBs to gather data from the **NP**. Since the NP database may receive large amounts of requests the database tier can also be replicated for redundancy.

The **DSP**:

Again, servlets and EJBs will abstract away connections and data source-specific code, presenting only an API to the other components.

The **EScore**:

The **EScore** module will interact directly with the **Xactans** server, using APIs and will determine the score of each document that will be relayed back to the **ESpider** module. Should the amount of processing power needed by the EScore module become to large for the web application to handle, some jobs will be placed in a queue which will be executed at a later time. A system notification will be issued to the user and the system administrator. The configuration of the web application will be on a per deployment basis so that it can be tweaked by an administrator for optimum performance.

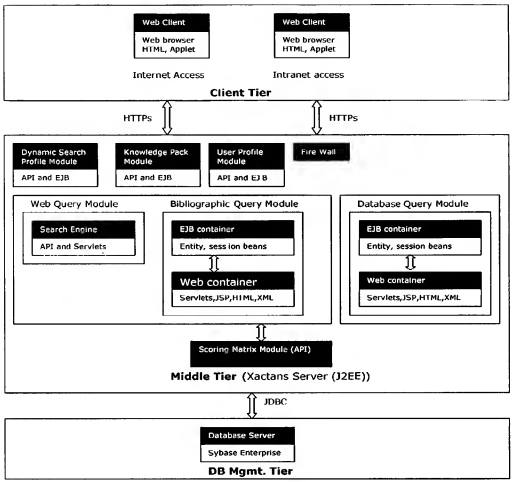

Figure 6. System Architecture Design of the Xactans

Data Storage:

The backend tier is the Database Management System (DBMS), which interacts with the middle tier by employing an internal API over a specific database connection (Java Database Connectivity (JDBC) technology for relational databases). Using object-oriented logic the functionality of the database entities are abstracted into interfaces that are then implemented for a specific database implementation on an enterprise server. By default these system databases will only store documents once and provide links to the original data source. Copies will be made only on demand. Account profile data is specific to each user and only editable by the user or an administrator. The database system provides the ability to store disparate data types.

Software Development:

We developed a search analysis tool: **ESpider**, that provides users with customized citation retrieval from a database of 34 on-line-Journals and reports the presence of ATCC cell lines and clones within these documents. These functionalities were required in order to comply with an NCRR-funded initiative.

Xactans is comprised of three distinct modules responsible for: 1) web query, 2) Journal article database query, and 3) specialized database query. The query and document processing features of the first two modules will be very similar, as discussed above. The database integration capability will be unique. We

will use PubMed Central as the source of retrieved data and subject it to our document processing modules. For internet-accessible source documents, we will focus on domestic (.edu) domain sites.

As shown (figure 7), we plan to develop an integrated molecular biology database query module where users will have an option of directly searching the databases of their choice or re-using a previously entered query once document processing is completed (Figure 1).

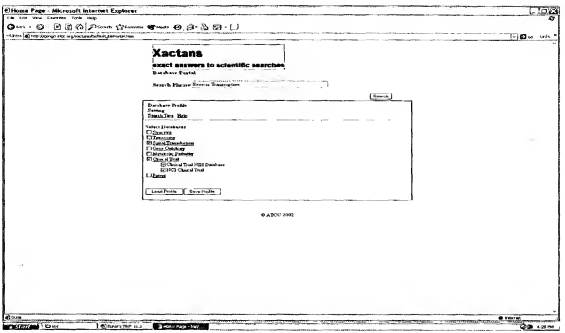

Figure 7. Snapshot of the database selection and query page.

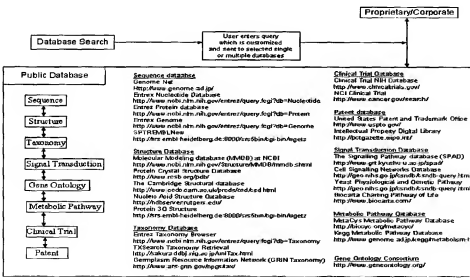

Table 1: A list of source databases, in different categories, that will be integrated in the database search module of **Xactans**.

Initially, we plan to integrate specialized sequence databases related to primary amino acids and nucleotides, metabolic pathways, clinical trials, taxonomy and the USPTO. Furthermore, ESpider systems can be developed to better suit user needs regarding proprietary and corporate LAN-accessible databases. In addition, a system will be developed to ignore dead-links, and incorporate new links of interest.

Output Processing:

Retrieved documents are processed to extract the data relevant to answer the query. The **ESpider** module is able to analyze each document entirely and calculate the frequency of occurrence of search terms separately and in combination in different parts of the document. Our present version (of **ESpider**) calculates the frequency of user's defined terms separately in the document, and locates the most frequently repeated terms/phrases with number of repeats. The following is an example of the **ESpider** program flow (truncated) for calculating basic parameters for the **EScore** module:

```
sub _repair_info_for_output
{
        my $self  = shift;
        my $htmlString      = shift;
        my $_returnVal      = '';
        $_returnVal = 'url:             ' . $self->{'url'} . "\n";
        $_returnVal .= 'Total Match:    ' . ${ $self->_count_number_of_words_occur_in_html_page( $htmlString ) } .
"\n";
        $_returnVal .= 'Total Words (Est): ' . ${ $self->_count_number_of_words_which_match_the_parse_search(
$htmlString ) } . "\n";
        $_returnVal .= 'Title:          ' . ${ $self->_retrieve_document_title( $htmlString ) } . "\n";
        $_returnVal .= 'Desc:           ' . ${ $self->_retrieve_document_meta_description( $htmlString ) } . "\n";
        $_returnVal .= 'Keywords        ' . ${ $self->_retrieve_document_meta_keywords( $htmlString ) } . "\n";
        $_returnVal .= 'Date            ' . ${ $self->_retrieve_document_last_update( $htmlString ) } . "\n";
        $_returnVal .= 'Author          ' . ${ $self->_retrieve_document_author( $htmlString ) } . "\n";
        $_returnVal .= 'Lang            ' . ${ $self->_retrieve_document_lang( $htmlString ) } . "\n";
        $_returnVal .= 'Most Occur Words: ' . join (', ', @{ $self->{'MostOccurWords'} }) . "\n";
        $_returnVal .= '##########################################' . "\n\n";
        $self->_empty_current_values();

return \$_returnVal;
}
sub _count_number_of_words_occur_in_html_page
{
        my $self            = shift;
        my $_thisHTMLCode       = shift;
#       Calculate number of words inside passing HTML
code
        ....
}
sub _count_number_of_words_which_match_the_parse
_search
{
        my $self            = shift;
        my $_thisHTMLCode       = shift;
#       Get number of words that match set up search
term
        ....
}
sub _retrieve_document_title
{
        my $self            = shift;
        my $_thisHTMLCode       = shift;
#       Get document title
        ....
}
sub _retrieve_document_meta_description
{
        my $self            = shift;
        my $_thisHTMLCode       = shift;
#       retrieve document description if existing
        ....
}
#       retrieve document description if existing
        ....
}
sub _retrieve_document_meta_keywords
{
        my $self            = shift;
        my $_thisHTMLCode       = shift;
#       Retrieve document keywords if existing
        ....
}
sub _retrieve_document_last_update
{
        my $self            = shift;
        my $_thisHTMLCode       = shift;
#       Retrieve document last update
        ....
}
sub _retrieve_document_author
{
        my $self            = shift;
        my $_thisHTMLCode       = shift;
#       Retrieve author name
        ....
}
sub _retrieve_document_lang
{
        my $self            = shift;
        my $_thisHTMLCode       = shift;
#       Retrieve language
        ....
}
```

Table 2. A sample of PERL programming code for **ESpider** that operates on extracting the frequency of scientific term match in a retrieved document.

There will be a constant upgrade feature that would build a comprehensive and up-to-date Index database. For example, we used **ESpider** to process the results of a search of the Proceedings of the National Academy of Sciences (PNAS) journal database based on a simple query of "reverse", "transcriptase", and "crystal structure". Our objective was to find an article on structure-function and drug design for HIV-1 RT and HIV-2 RT enzymes. As indicated above, a series of returned values is obtained and displayed as follows:

```
url:http//www.pnas.org/cgi/content/full/96/18/10027?maxtoshow=&HITS=10&hits=10&RESULTFORM
AT=&fulltext=reverse+transcriptase+crystal+structure&searchid=1035920455241_7999&stored_s
earch=&FIRSTINDEX=0
Total Match:          16
Total Words (Est): 6942
Title:                Lamivudine (3TC) resistance in HIV-1 reverse transcriptase involves
steric hindrance with β-branched amino acids. PNAS -- Sarafianos et al. 96 (18): 10027
Date                  Wed, 30 Oct 2002 15:54:16 GMT
Proximal Scientific Terms (Frequency):  RT(57), HIV-1(45), 3TC(44), ABSTRACT(44),
RESISTANCE(43), M184I(35), WILD-TYPE(33), COMPLEX(32), 3TCTP(29), TEXT(26), VIRUS(26),
POSITION(26), MUTANT(25), MODEL(25), J(22), POLYMERASE(22), Å(21), INHIBITORS(21),
STRUCTURE(20), DNA(20)
```

The proximal scientific term frequency value is obtained via the $_returnVal .= 'Most Occur Words: ' . join (', ', @{ $self->{'MostOccurWords'} }) . "\n"; subroutine.

```
url:http://www.pnas.org/cgi/content/full/99/22/14410?maxtoshow=&HITS=10&hits=10&RESULTFOR
MAT=&fulltext=reverse+transcriptase+crystal+structure&searchid=1035920455241_7999&stored_
search=&FIRSTINDEX=0
Total Match:          27
Total Words (Est): 5459
Title:                Structure of HIV-2 reverse transcriptase at 2.3 Å resolution and the
mechanism of resistance to nonnucleoside inhibitors.  PNAS -- Ren et al. 99 (22): 14410
Date                  Wed, 30 Oct 2002 16:01:56 GMT
Proximal Scientific Terms (Frequency):  RT(132), HIV-2(89), HIV-1(71), NNRTI(35),
STRUCTURE(29), RESIDUES(22), NNRTIS(21), BINDING(20), SITE(18), POCKET(17), SUBUNIT(17),
RESISTANCE(15), CRYSTAL(14), REGION(14), SIDE(14), Å(14), STAMMERS(12), P68(12),
INHIBITORS(12), DATA(12)
##############################################
```

The current output is frequency of match based rather than weight based. In another example we performed a full text search of PubMed Central journal articles for "reverse transcriptase" and found 1881 hits. Processed results for five returned documents:

```
http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=128826
Total Match:          2 (reverse transcriptase)
Total Words (Est): 5276
Title:                Expression of human beta-defensins 1 and 2 in kidneys with chronic
bacterial infection
Date                  Wed, 13 Nov 2002 16:12:00 GMT
Most Occur Words:  HBD-1(59), HBD-2(55), EXPRESSION(43), HUMAN(42), TISSUE(37), CELL(28),
RENAL(28), RT-PCR(25), RNA(21), GENE(20), ANTIMICROBIAL(19), KIDNEY(18), INFECTION(16)
##############################################
http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=126235
Total Match:          1 (reverse transcriptase)
Total Words (Est): 6826
Title:                Identification of Schistosoma mansoni gender-associated gene
transcripts by cDNA microarray profiling
Date                  Wed, 13 Nov 2002 16:19:14 GMT
Most Occur Words:  CDNA(100), MANSONI(53), MICROARRAY(75), GENE(51), SCHISTOSOMA(43),
FEMALE(43), SCHISTOSOME(42), EXPRESSION(41), MALE(31), ADULT(30), FIGURE(27),
PARASITOL(23), TRANSCRIPTS(21), CLONES(39), HYBRIDIZATION(20)
##############################################
```

```
http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=126239
Total Match:        1 (reverse transcriptase)
Total Words (Est): 6788
Title:              Accurate normalization of real-time quantitative RT-PCR data by
geometric averaging of multiple internal control genes
Date                Wed, 13 Nov 2002 16:22:18 GMT
Most Occur Words:  GENES(219), CONTROL(126), EXPRESSION(85), NORMALIZATION(81),
VARIATION(41), HOUSEKEEPING(31), TISSUES(52), STABLE(24), RT-PCR(24), RNA(23)
##############################################
http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=117927
Total Match:        15 (reverse transcriptase)
Total Words (Est): 9067
Title:              Downstream E-Boxmediated Regulation of the Human Telomerase Reverse
Transcriptase (hTERT) Gene Transcription: Evidence for an Endogenous Mechanism of
Transcriptional Repression
Date                Wed, 13 Nov 2002 16:31:07 GMT
Most Occur Words:  HTERT(126), CELLS(160), ET(90), RCC23+3(76), E-BOX(74), RCC23(69),
PROMOTER(67), HUMAN(67), TELOMERASE(61), TRANSCRIPTION(59), GENE(53), DOWNSTREAM(45),
CANCER(43), REPRESSION(38), CHROMOSOME(37)
##############################################
http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=60646
Total Match:        12 (reverse transcriptase)
Total Words (Est): 3663
Title:              Analysis of HIV-1 drug resistant mutations by line probe assay and
direct sequencing in a cohort of therapy naive HIV-1 infected Italian patients
Date                Wed, 13 Nov 2002 16:37:37 GMT
Most Occur Words:  RESISTANCE(46), MUTATIONS(32), HIV-1(31), PATIENTS(27), DRUG(26),
ASSAY(25), VIRUS(23), VIRAL(20), LIPA(18), THERAPY(17), ANTIRETROVIRAL(17), PROTEASE(17),
HIV(16), ASSOCIATED(15), PRESENCE(14), RT(13), NAIVE(13)
##############################################
```

This sample output offers a unique opportunity for users to be able to scan a document without reading it. With minimal training, users will become accustomed to assessing the relevance of retrieved documents. In the above example the key word "reverse transcriptase" only appears once for two of the sample output documents, thus classifying these articles as less relevant in this case.

This important key word frequency information becomes available due to development of the **ESpider** document- processing module. Moreover, we plan to expand **ESpider** to include processing and extraction of data based on the following parameters, which will be transferred to the **EScore** module to assign relevance rank based on:

i. Frequency of user's chosen keyword or synonyms
ii. Presence/Proximity assessment of keywords and related words in same sentence
iii. Presence/Proximity assessment of keywords and related words in the same paragraph
iv. Frequency of **NP** terms in the entire document
v. User's defined phrases
vi. Location of above terms in the document (initially we will focus on title, abstract, body of the text).

In the future we plan to parse output documents and display the results summary in an easily readable form by employing full information extraction (IE) and knowledge discovery methods. User input query terms will be highlighted. This will be an iterative process and we will optimize **ESpider**:**EScore** interaction to achieve our goal of establishing parameters that interconnect query terms to the degree of user interest in a retrieved document.

The **EScore** System:

Query terms may be associated with synonymous words, related words, scientific fields, etc., and thus relationships of different term weights may be assessed. **EScore** is an evolving algorithm that will create a scheme of relationship scoring through a network of relational matrices (Burden et al., '01; Heath et al., '02), in order to determine the degree of usefulness of web page information with respect to a searched word. Each

matrix is used to score data based on particular criteria, such as proximity to the query term and the number of exact matches, proximity and frequency of synonyms, the location of these terms in the document—i.e. in the title, abstract or body of the text. The results are presented to the user in descending score order.

In addition, a matrix might show a relationship between a number of search word synonyms and its' related words. There will be matrices with several dimensions, where a dimension equates to multiple relationships between terms of interest. A number of searched words found in the abstract may be associated with a number of NP terms found in the same section (Figure 6), such that an instance of the matrix element would produce a specific score.

The previous example would also be associated with a number of related words in the same paragraph, yielding a three dimensional matrix with three relationships. A software routine or routines would run parameters against available matrices to come up with a partial score for each matrix. The total score of the matrix is always constant, but element scores within any matrix are dynamic statistical probabilities of occurrences and change through a feedback mechanism. The presented approach is a slight modification of a Markov Model (Mathews et al., '99; Durbin et al., '98) shown here:

$$P(\text{total})=P(x_1)P(x_2|x_1)P(x_3|x_2)\ldots P(x_L|x_{L-1})$$

where P(total) is the product of individual probabilities P(x) for a total of L number of instances.
In Markov's model the total probability is the product of individual probabilities where each unique occurrence in a system is associated with a specific probability that can be adjusted through training of a system. In our system, initial values in the matrix are arbitrary probabilities derived from an initial dataset. Software feedback will use the algorithm below to adjust individual probabilities in the matrix as more data is processed:

$$P(x_{cell})=(\text{adjustment}_{cell})^*(x_{cell}/\Sigma x_{cell})$$

All other matrices in the matrix network would have an associated score for the data at hand. The scores from each matrix would then be added up in the same way as impedance in an electrical circuit (Figure 9). A total score would represent a total assessment of all the relationships in our model. Based on user preferences on the search screen, a feedback mechanism would be able to weight adjust each matrix's output based on search profile input. This user induced feedback method, upon execution, will allow for fine-tuning of the selectivity of the query results.

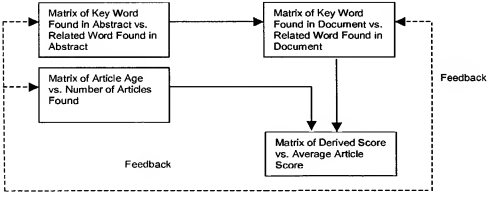

Figure 8. An example of a schema used to derive a score for each searched article

Matrices positioned in series would require an input from a previous matrix's output, thus establishing a sequential relationship. Parallel matrices would be independent of each other's output and could process information concurrently. The scoring process could be distributed by using multithreaded logic of parallel processing as opposed to sequential processing of serial logic data. As stated above, adding matrix scores in

parallel would be different than adding scores in series, where the serial dependent relationship, consisting of more than one dependent step, produces a higher total score than for independent matrices in parallel.

A software array, which can be multidimensional, would represent each matrix, and thus the relationship model can be easily modified in terms of software development and updates (Preiss, '00). During execution, array data that represents a score for a relational instance could be adjusted through a software feedback mechanism. Java is a powerful programming language for working with arrays and matrices, since many methods have already been implemented that would simplify the development process. Java is also operating system agnostic and thus allows for greater flexibility for development and execution.

In a more specific scenario of how a program would rate web pages, we turn to the parameters that would get passed between different functions or classes. A specific number would be generated for each parameter of interest during a parsing of each retrieved document. We could also have additional parameters of importance for each page, such as article age, overall number of articles found and publishing entity of an article. It may not be known until the development and testing stage of **Xactans**, which parameters would carry more weight in producing better results, and it is important to structure the program such that it can be easily altered and parameter structures modified. Scores for all matrices would then be added up to generate a total score. The total score of perceived relevance that is generated along with the web page reference would be passed on to **ESpider**, which would process and present results to the end user.

Possible Problems and their Solutions

1. Dead and Nonfunctional Web Links: **Xactans** will employ public domain databases and websites whose uptime and maintenance is not guaranteed. We have installed Xenu's Link Sleuth (TM) on our server to check Web sites for broken links. Link verification for "normal" links, images, frames, plug-ins, backgrounds, local image maps, style sheets, scripts and java applets is routine. In addition, for the ATCC MR4 website which is a resource for malaria-related research protocols, and reagents, (http://www.malaria.mr4.org/) we have developed an automated error monitoring system that helps us rectify problems quickly. Protocols and sources of reagents can change. In the event that a link to a Protocol site is no longer available, a message is generated and sent to our webmaster, who maintains our MR4 website.
2. Network Security and User Profiles: We have employed a Raptor firewall system to protect the data on our server from outside hackers. User access will be authenticated through login and password by using a Secure Socket Layer (SSL) for secure communication between our server and user interface. All communication will be encrypted.
3. **NP** authentication: Scientific terms and searched keywords will be stored as anonymous holding files without links to user information. In order to improve the quality of DSPs, experts in their fields will check each NP element and definition. During phase two, we will form two advisory committees (Scientific and Technical) with experts from different life sciences disciplines, the pharmaceutical industry and library management.
4. Data Processing and Expandability: The importation of entire files and document processing is complex and requires extensive amounts of time depending on the level of internet traffic, document originating server processing speed, and data structure. Our strategy will be to restrict our searches to one layer deep and process the first 15-20% of documents returned to ensure the validity of the search system. We will cache information retrieved in the index file for future use and slowly start building our index database. As the **Xactans** project proceeds we will gradually open and expand the search criteria. Our aim would be to complete our index database once and then proceed with updating information from existing sites and adding new information from any newly developed/created sites. This will accelerate the search, retrieval and data processing process. During phase one we will formulate a working estimate of the number of gigabytes of scientific data currently available, and its growth rate. Thus, we will be better positioned to establish the needed infrastructure in phase two.

Project Milestones:

Phase I:

1. Develop the NP in a relational database setting (6 months).
2. Apply **Xactans** to .edu domain sites and journal articles, available on PubMed Central, for search, retrieval and document processing (6-9 months).
3. Incorporate the EScore module (6-9 months). Write code for six scoring matrices during phase one, we will limit scoring matrices such that attributed fields for titles, abstracts and text bodies will be attached to query search- and related terms (8 months).
4. Integration of the **ESpider** database search module (6 months).

References:

Baxevanis, A.D. (2002) The Molecular Biology Database Collection: 2002 update. Nucleic Acid Research 30, 1-12.

Burden, R.L., J. Douglas Faires (2001Numerical Analysis, 7th Edition, Wadsworth Group.

Durbin, R., S. Eddy, A. Krogh, G. Mitchison, Ch. 3, Biological Sequence Analysis, Cambridge University Press, 1998

Heath, M.T. (2002) Scientific Computing An Introductory Survey, Second Edition, McGraw-Hill.

Lawrence, S. and Giles, C.L. (1998) Context and Page Analysis for Improved Web Search. IEEE Internet Computing 2, 38-46.

Mathews, J.H, Kurtis D. Fink, (1999) Numerical Methods Using Matlab, Third Edition, Prentice-Hall pp 579-590.

Preiss, B.R., Ch 4, Data Structures And Algorithms With Object-Oriented Design Patterns In Java, John Wiley & Sons, Inc., 2000

Stein, L. (2002) Creating a Bioinformatics Nation. Nature 417, 117-123.

Voss, D. (2001) Better Searching Through Science. Science 293, 2024-2026.

Wilkinson, S.L. (2002) A Guide to Digital Literature. Chemical and Engineering News, January 7, 2002 pp 30-33.

http://Google.com
http://Scirus.com
http://www.cas.org/SCIFINDER/scicover2.html
http://www.search4science.com/
http://umlsks1.nlm.nih.gov/kss/servlet/Turbine;jsessionid=4xhdg2xqp1
http://ncbi.nih.gov/
http://www.malaria.mr4.org/
http://stemcells.atcc.org/
http://searchenginewatch.com/